

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images,
please do not report the images to the
Image Problem Mailbox.**

#2

Attorney Docket: 381NP/47598
PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE



Applicant: FUMIO NARISAWA, ET AL
Serial No.: NOT YET ASSIGNED
Filed: FEBRUARY 22, 1999
Title: OBJECT-ORIENTED OPTIMIZATION CODE GENERATOR
AND A METHOD THEREFOR

CLAIM FOR PRIORITY UNDER 35 U.S.C. §119

Box PATENT APPLICATION

February 22, 1999

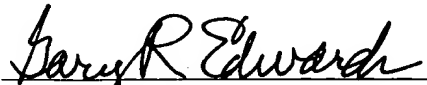
Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

The benefit of the filing date of prior foreign application No. 10 038329 on February 20, 1998, is hereby requested and the right of priority under 35 U.S.C. §119 is hereby claimed.

In support of this claim, filed herewith is a certified copy of the original foreign application.

Respectfully submitted,



Gary R. Edwards
Registration No. 31,824

EVENSON, McKEOWN, EDWARDS
& LENAHA, P.L.L.C.
1200 G Street, N.W., Suite 700
Washington, DC 20005
Telephone No.: (202) 628-8800
Facsimile No.: (202) 628-8844
GRE:kms

日 本 国 特 許 庁
PATENT OFFICE
JAPANESE GOVERNMENT



別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日

Date of Application:

1 9 9 8 年 2 月 2 0 日

出 願 番 号

Application Number:

平成 1 0 年 特 許 願 第 0 3 8 3 2 9 号

出 願 人

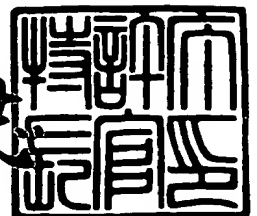
Applicant (s):

株式会社日立製作所

1 9 9 8 年 1 0 月 2 3 日

特 許 庁 長 官
Commissioner,
Patent Office

伴 佐 山 建 志



出証番号 出証特平 1 0 - 3 0 8 5 2 6 3

【書類名】 特許願

【整理番号】 1197025781

【提出日】 平成10年 2月20日

【あて先】 特許庁長官 殿

【国際特許分類】 G06F 9/44

【発明の名称】 オブジェクト指向最適化コード生成装置及び方法

【請求項の数】 10

【発明者】

【住所又は居所】 茨城県日立市大みか町七丁目1番1号
株式会社 日立製作所 日立研究所内

【氏名】 成沢 文雄

【発明者】

【住所又は居所】 茨城県日立市大みか町七丁目1番1号
株式会社 日立製作所 日立研究所内

【氏名】 納谷 英光

【発明者】

【住所又は居所】 茨城県日立市大みか町七丁目1番1号
株式会社 日立製作所 日立研究所内

【氏名】 横山 孝典

【発明者】

【住所又は居所】 茨城県ひたちなか市大字高場2520番地
株式会社 日立製作所 自動車機器事業部内

【氏名】 大川 圭一朗

【特許出願人】

【識別番号】 000005108

【氏名又は名称】 株式会社 日立製作所

【代理人】

【識別番号】 100068504

【弁理士】

【氏名又は名称】 小川 勝男

【電話番号】 03-3212-1111

【手数料の表示】

【予納台帳番号】 013088

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9003094

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 オブジェクト指向最適化コード生成装置及び方法

【特許請求の範囲】

【請求項1】

入力されたオブジェクト指向に基づく仕様書を解析し、仕様情報を抽出する仕様解析手段と、該仕様解析手段により得た前記仕様情報を予め定められた機能排除規則に従って、オブジェクト指向の機能のうち不要な機能を排除したプログラム情報を生成する機能排除手段と、該機能排除手段により得た前記プログラム情報に基づいてコードを生成するコード生成手段とを有することを特徴とするソフトウェア生成装置。

【請求項2】

請求項1のソフトウェア生成装置において、

前記機能排除手段は、前記機能排除規則により仮想関数の機能を排除することを特徴とするソフトウェア生成装置。

【請求項3】

請求項1のソフトウェア生成装置において、

前記機能排除手段は、前記機能排除規則によりインスタンスの動的生成の機能を排除することを特徴とするソフトウェア生成装置。

【請求項4】

請求項1のソフトウェア生成装置において、

前記機能排除規則は、オブジェクト名、メソッド名からなる入力パターンと、オブジェクトの機能と、該オブジェクトの機能の使用、不使用と、該使用に対する出力コードの生成パターン及び不使用に対する出力コードの生成パターンから構成されていることを特徴とするソフトウェア生成装置。

【請求項5】

図形情報として記述された仕様書及び使用するオブジェクト指向の機能を選択するための入力手段と、

前記入力手段により入力された仕様書を解析する解析手段と、

前記解析手段により得られた解析結果を前記選択された機能に基づいてコード

を生成するためのパターン情報を出力する機能選択手段と、

上記機能選択手段から出力されたパターン情報に基づいて、前記解析された仕様書のプログラムコードを生成するコード生成部とを有することを特徴とするソフトウェア生成装置。

【請求項 6】

請求項 5 のソフトウェア生成装置において、

前記機能選択手段は、仮想関数の機能のみを選択し、前記コード生成部で上記仮想関数の機能を使ったコードを生成することを特徴とするソフトウェア生成装置。

【請求項 7】

請求項 5 のソフトウェア生成装置において、

前記機能選択手段は、インスタンスの動的生成の機能のみを選択し、前記コード生成部で上記インスタンスの動的生成の機能を使ったコードを生成することを特徴とするソフトウェア生成装置。

【請求項 8】

入力されたオブジェクト指向に基づく仕様書を解析し、仕様情報を抽出する仕様解析手段と、

該仕様情報からオブジェクト指向機能の仕様状況を表示する解析結果表示手段と、

使用するオブジェクト指向の機能を選択するための入力手段と、

上記入力手段によって選択された機能を記憶する機能記憶手段と、

前記仕様解析手段により得た前記仕様情報を前記記憶手段に記憶した前記選択された機能に従って、該機能を使用したプログラム情報を生成するプログラム情報生成手段と、

前記プログラム生成手段により得た前記プログラム情報に基づいてコードを生成するコード生成手段とを有することを特徴とするソフトウェア生成装置。

【請求項 9】

請求項 8 のソフトウェア生成装置において、

前記解析結果表示手段は、それぞれのメソッドのうち用いられていないメソッ

ドを表示することを特徴とするソフトウェア生成装置。

【請求項 10】

入力されたオブジェクト指向に基づく仕様書を解析し、予め定められた機能排除規則に従って、オブジェクト指向の機能のうち不要な機能を排除したプログラム情報を生成し、前記プログラム情報に基づいて前記解析された仕様書のコードを生成することを特徴とするソフトウェア生成方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明はソフトウェアの自動生成装置に係り、特にオブジェクト指向仕様書から、コード効率の良いコードを生成するソフトウェア生成装置に関する。

【0002】

【従来の技術】

コード生成装置は、オブジェクト指向のソフトウェア記述からソフトウェアを生成するものである。これは、一般に、オペレータの記述した仕様書を入力装置から入力し、その仕様を解析し、プログラムコードを生成し出力するものである。

【0003】

しかしながらこのようにオペレータの記述した仕様書に基づいて単にコード生成を行うと目的とするコードが非常に大きくなり、大きなメモリ容量が必要になるといった問題が生ずる。

【0004】

この問題を解決するものとして、特開平6-266562号公報には、目的コードの大きさを削減する方式が記述されている。これは、オペレータの記述した仕様書の構文意味解析部において、クラス宣言の意味解析処理時にメソッドの所属するクラス名と入力ファイル名とを比較し、一致すればメソッドアドレス格納テーブルの实体および宣言を出力コード内に生成し、一致しなければメソッドアドレス格納テーブルの宣言のみを出力コード内に生成するものである。

【0005】

【発明が解決しようとする課題】

しかしながら、上述のようなコード生成方式では、オブジェクト指向のプログラムのコードを生成する場合のオブジェクト指向の機能については何ら考慮されていない。

【0006】

つまり、オブジェクト指向の仕様書からコードを生成する場合、不要となるオブジェクト指向の機能についてもコードを生成してしまう。

【0007】

特に組み込み制御システムのソフトウェアに用いるには出力されたコードが大きくなり、それを搭載するためにメモリ容量のおおきい装置を用いることになる、コスト増大をまねくといった問題がある。

【0008】

本発明は、上記問題点に鑑みて、必要なメモリ容量を増加させることなく組み込み制御システムに適用可能なコードの最適化を行うコード生成装置を提供することを目的とする。

【0009】

【課題を解決するための手段】

上記目的は、入力されたオブジェクト指向に基づく仕様書を解析し、仕様情報を抽出する仕様解析手段と、仕様解析手段により得た仕様情報を予め定められた機能排除規則に従って、オブジェクト指向の機能のうち不要な機能を排除したプログラム情報を生成する機能排除手段と、機能排除手段により得たプログラム情報に基づいてコードを生成するコード生成手段とを有することにより達成することができる。

【0010】

【発明の実施の形態】

図1は、コード生成装置の全体構成を示したものである。コード生成装置はソフトウェア仕様や最適なコードを生成するための条件である機能選択項目を入力するためのキーボードやマウスなどの入力装置101、入力装置101のキーボ

ード等から入力されたソフトウェア仕様や機能選択項目、生成されたコードを表示するCRTなどの表示装置102、フロッピーディスク等の携帯型の記憶装置に記憶されたソフトウェア仕様や機能選択項目を読み出したり、生成されたコードを書き込むための書込／読出装置103、入力装置101又は書込／読出装置103から入力されたソフトウェア仕様や機能選択項目を記憶するための記憶装置104、入力装置101、記憶装置104から送られてくるソフトウェア仕様と機能選択項目に基づいて最適なコードを生成するための処理装置105から構成されている。

【0011】

また、処理装置105は入力されたソフトウェア仕様に対して字句解析や文法解析を行う仕様解析部106、入力された機能選択項目に基づいて使用しない機能を排除するオブジェクト指向機能排除部107、オブジェクト指向機能排除部107で使用しないとされた機能を排除して仕様解析部106で字句解析、文法解析を行ったソフトウェア仕様からコードを生成するコード生成部108、入力装置101からソフトウェア仕様及び機能選択項目を入力するために表示装置102の画面を制御する入力画面制御部109、各部の制御を行う全体制御部110から構成されている。更に入力画面制御部109はクラスインターフェイス記述部111、処理詳細記述部112、機能選択項目記述部113から構成されている。尚、コード生成部108で生成されたコードは最終的に表示装置102に表示されたり、書込／読出装置103からフロッピーディスク等の携帯型の記憶装置に記憶される。

【0012】

図2は、処理装置105の全体制御部110の動作を示したフローチャートである。本処理装置105は、コードを生成するソフトウェア仕様や機能選択項目を入力する入力モードと、入力されたソフトウェア仕様からコードを生成するコード生成モードがある。入力装置101からのオペレータの指示により入力モードが選択された場合、全体制御部110は入力画面制御部109を起動する。起動された入力画面制御部109は、クラスインターフェイス記述部111によりオペレータからのソフトウェア仕様の一形態であるモデル図の入力を受け付け、

記憶装置 104 にこのモデル図を格納する。

【0013】

次に入力画面制御部 109 の処理詳細記述部 112 を起動させ、オペレータがソフトウェア仕様の一形態である動作図の入力を受け付け、この動作図を記憶装置 104 に記憶する。

【0014】

最後に、機能選択項目記述部 113 を起動させ、入力されたソフトウェア仕様からコードを生成するための条件である機能選択項目の入力を受け付け、記憶装置 104 に記憶する。

【0015】

一方、オペレータによりコード生成モードが選択された場合、全体制御部 110 は、記憶装置 104 に記憶されたソフトウェア仕様であるモデル図、動作図を読みだし、仕様解析部 106 を起動させて、字句解析や文法解析を行う。次にオブジェクト指向機能排除部 107 を起動させて、記憶装置 104 に記憶された機能選択項目に基づいて、仕様しない機能を決定する。次に、コード生成部 108 を起動させて、仕様解析部 106 で字句解析、文法解析が行われたソフトウェア仕様とオブジェクト指向機能排除部 107 で決定された出力コードパターンに基づいてコード生成を行う。尚、生成されたコードは、オペレータの指示により表示装置 102 に表示したり、書込／読出装置 103 により、フロッピーディスクに書き込まれる出力処理を行う。

【0016】

以下、オブジェクト指向の分析・設計において O O I E と呼ばれる手法を用いて各処理を具体的に説明する。

【0017】

まず、オペレータによって入力モードが選択された場合について説明する。入力モードの選択により、全体制御部 110 はクラスインターフェイス記述部 111 を起動する。これによりオペレータはソフトウェア仕様として図 3 に示すモデル図を入力装置 101 のキーボードやマウスを使って入力する。なお、図 3 に示したモデル図は自動車のエンジン制御などの制御の一部であるセンサからの入力値

に対する処理を示したものである。

【0018】

図3に示したモデル図は、オブジェクト指向のプログラミングにおいて、プログラムの部品となるオブジェクトの構成を記述したものである（なお、ここで示したモデル図はオブジェクト指向の分野において、クラス図、オブジェクト図などと呼ばれることもある）。

【0019】

モデル図は静的な情報を詳細に記述したものであり、3つの欄から構成されている。第1欄301には、クラスの名称に相当する変数の名前や型を、第2欄302にはデータの構造体の定義を、第3欄303には行う処理の名称と引数型をそれぞれ記述する。

【0020】

図3に示したモデル図は、クラスの名称が第1欄301に記述されているように「センサ」であることを示している。また、このクラスが持つデータは第2欄302に記述されているように「A/D変換値」、「更新値」、「前回更新値」であり、それぞれ「A/D変換値」は「unsigned char」、「更新値」は「signed short」、「前回更新値」は「signed short」の型で定義されていることを示している。更に、行う処理を示すメソッドは第3欄303に記述されているように「判定（）」、「更新（）」であり、「判定」は引数をとらず、戻り値の型が「boolean(真理値）」、「更新（）」は引数をとらず、戻り値も持たないことを示している。このようにオペレータによって記述されたモデル図の入力が終了するとクラスインターフェイス記述部111はモデル図を記憶装置104に記憶させる。

【0021】

次に、全体制御部110は、処理詳細記述部112を起動させ、オペレータは図4に示すような動作図をマウス、キーボード等を使って入力する。動作図とはそれぞれのクラスに属するオブジェクトのもつ処理を示すメソッドの詳細を記述したものである（なお、オブジェクト指向において、動作図の他に、コントロールチャート、状態遷移図などと呼ばれることもある）。図4は、図3の第3欄に

示したセンサの処理を記述したものであり、「判定 ()」401でセンサからの入力であるA/D変換値が予め定めた条件を満足するかを判定し、条件を満足する (YES) 場合には「更新」を行い、条件を満足しない (NO) 場合には「更新値=前回更新値」403とする処理を行うことを示している。オペレータにより、入力された動作図は、記憶装置104に記憶される。

【0022】

次に全体制御部110により、機能選択項目記述部113が起動され出力装置102には機能選択の入力画面が表示される。機能選択とは、“インスタンスの動的生成”，“仮想関数”などのオブジェクト指向の機能をコード生成の際に使用するか、使用しないかを選択するためのものである。

【0023】

図5に機能選択項目の入力画面の表示例を示す。これは、オペレータによってオブジェクト指向の“インスタンスの動的生成”の機能選択項目が選択された場合の入力画面を示したものである。この機能選択項目は、入力パターン501，使用機能項目502，設定選択肢503，出力コードパターン504の4つの項目から構成されている。入力パターンは、対象となるプログラムのオブジェクト名，メソッド名を入力するためのものであり、使用機能項目502はオブジェクトプログラムの機能を示す部分である。設定選択肢503は、使用機能項目502に示された機能を“使う／使わない”を選択するものであり、出力コードパターン504は、設定選択肢503の選択肢に対する出力コードパターンが示されている。つまり、設定選択肢503の“使う”について“オブジェクト名，メソッド名 (引数)”という生成パターン情報が対応し、“使わない”について“オブジェクト名__メソッド名 (引数)”という生成パターン情報が対応している。尚、“オブジェクト名__メソッド名 (引数)”は部品名を表すものであり、オブジェクト指向プログラミング言語の“インスタンスの動的生成”を使わずにコードを生成することを意味する。

【0024】

この機能選択項目は、予め使用機能選択項目502，出力コードパターン504が入力されている。従って、オペレータは、入力パターン501，設定選択肢

503を入力することにより簡単にオブジェクト指向の機能を使う／使わないを選択することができる。オペレータによって、決定された機能選択項目は、記憶装置104に格納される。

【0025】

以上、説明したように入力モードでは、ソフトウェア仕様と機能選択項目がオペレータによって入力される。

【0026】

次にオペレータによって、コード生成モードが選択された場合について説明する。

【0027】

オペレータによって、コード生成モードが選択されると、全体制御部110は、仕様解析部106を起動する。

【0028】

仕様解析部106で字句解析及び構文解析が行われ、図3、図4に示したように絵情報として入力されたソフトウェア仕様の情報から切り出した文字列と、この文字列の種類とを出力する。

【0029】

図6は、仕様解析部106が行う処理のフローチャートを示したものである。仕様解析部106は、ステップ601でオペレータによって入力されたソフトウェア仕様を記憶装置104から読み出す。絵情報として入力されたソフトウェア仕様の情報は、ステップ602で予め記憶装置104に記憶した仕様記述文法とのマッチングを行う。このマッチングについては、“lex & yacc プログラミング”，John R. Levine/Tony Mason/Doug Brown 共著，1996.6.21”に記載されているlex と yacc を使って行うものであり、BNF（バックス・ナウア記法）などで記述された記述構文と入力されたソフトウェア仕様とがマッチしているか判定する。判定の結果、仕様記述文法とマッチしなければステップ604で“error”を表示装置102に出力し、処理を終了する。判定の結果、仕様記述文法とマッチした場合にはステップ203で、入力されたソフトウェア仕様の文字列と、この文字列の変数名とを組として出力する。

【0030】

例えば、オペレータによって入力された図6に示したソフトウェア仕様の情報は、文字列“センサ”，変数名“変数1”文字列“A/D変換値”，変数名“変数2”，文字列“更新値”，変数名“変数3”と順次出力する。

【0031】

また、図4に示したソフトウェア仕様の情報は、角を丸めた四角形で囲んだものを“メソッド呼び出し”、四角形で囲んだものを“コマンド”と予め定義しておくことで、例えば401は属性が“メソッド呼び出し”、オブジェクト名が“センサ”，メソッド名が“判定”，引き数はない、と出力される。

【0032】

次に、全体制御部110はオブジェクト指向機能排除部107を起動する。

【0033】

オブジェクト指向機能排除部は、記憶装置104にオペレータによって入力された機能選択項目に基づいてコードパターンを出力する。

【0034】

図7にオブジェクト指向機能排除部107の処理のフローチャートを示す。

【0035】

ステップ701で仕様解析部106の出力である文法解析結果を入力する。次に、ステップ702でコードを生成するための条件である機能選択項目を文法解析結果から得られたオブジェクト名，メソッド名と機能選択項目の入力パターンに基づいて記憶装置104から読み出す。

【0036】

ステップ704では、ステップ702で記憶装置104からテーブルが読み出せなかった場合には標準の生成パターン情報を出力する。ステップ702で記憶装置104からテーブルを読み出せた場合には、読み出された機能選択項目に基づいて、機能を使用するか否かをステップ705で判定し、使用する場合には機能選択項目で定義された標準の生成パターン情報を出力し（ステップ707）不使用の場合には機能選択項目で定義された生成パターン情報を出力する（ステップ706）。

【0037】

次に全体制御部 110 はコード生成部 108 を起動し、コード生成を行う。

【0038】

図 8 は、コード生成部 108 の動作のフローチャートを示したものである。

【0039】

まず、ステップ 801 で、オブジェクト指向機能排除部 107 の出力である生成パターン情報を入力し、ステップ 802 で、この生成パターン情報と記憶装置 104 に記憶されているコード生成パターンに基づいてコード生成を行う。

【0040】

ここで、仕様解析部 106 から図 3、図 4 に示されたソフトウェア仕様とオブジェクト指向機能排除部 107 から“オブジェクト名，メソッド名（引数）”の生成パターン情報が入力された場合、コード生成部 108 は記憶装置 104 に記憶されているコード生成パターン情報に基づいてコードを生成する。

【0041】

つまり、記憶装置 104 にコード生成パターンとして、

“class” クラス名 “{”

型 1，データ 1；

型 2，データ 2；

・

・

・

返値型 1 メソッド 1（引数型 1）；

返値型 2 メソッド 2（引数型 2）；

・

・

・

“}”

・判定分

“if（“判定部”）”

条件成立時実行文；

“} else {”

条件不成立時実行文；

“} ”

と記憶されており、テーブルで“インスタンスの動的生成”が不使用と設定されている場合には、図9（a）に示すようにコードが生成される。そして、この結果をコンパイルすると図10（a）に示す実行コードが得られる。このコードは表示装置102に出力されたり、書込／読出装置103によりフロッピーディスクに記憶される。

【0042】

尚、図9（b）には“インスタンスの動的生成”を使用と設定した場合のコード生成を、図10（b）にはその実行コードを示している。

【0043】

これら2つを比較すると、動的生成機能を使う場合にはインスタンスを陽に指定するため、図中1001としてあらわされるインスタンスが明示され、インスタンスのアドレスをスタックに格納するためのコードが生成される。

【0044】

これに対して、動的生成を使わない場合には、1001で示したインスタンスを指定する部分がなく、1001に対応するコードは出力されない。このようにしてオブジェクト指向の機能を排除することによりコードサイズを削減することができる。

【0045】

オブジェクト指向の別の機能として“仮想関数”を行った場合について説明する。

【0046】

仮想関数は、ある親のクラスから複数の子のクラスを承継を用いて作成するときに、それぞれの子のクラスで実装が異なるときに、親のクラスで共通のインターフェイスを記述するものである。尚、“継承”とは、ある既存のクラスに対し、既存の性質を全て引き継いだ上で、差分を記述することで新たなクラスを作る

方法である。

【0047】

図11(a)は仮想関数を使うメソッドを指定したときに、オペレータが仮想関数を“使用”と設定し、既に述べた方法により生成したコードを実行形式にしたメモリ配置を示したものである。この図に示すようにプログラムコード部に関数“判定()”のコード1101が格納され、データ部に内部変数の領域1102と仮想関数“更新”の呼び出しのためのテーブル1103が格納される。仮想関数の仕組みを実現するのはデータ部に作られた仮想関数テーブル1101である。

図11(b)は仮想関数の機能を“不使用”と設定した場合の実行形式にコンパイルしたメモリ配置を示したものである。この場合、仮想関数は直接呼び出されるので、図11(a)の呼び出しテーブル1103に相当するものが不要となっている。

【0048】

このように、仮想関数やインスタンスの動的生成等のオブジェクト指向の機能を排除することで、オブジェクト指向の機能を実現するためのメカニズムを排除できる。また排除するには、オペレータには各オブジェクト指向のメカニズムに関する知識は不要で、“使用／不使用”を指定するだけで最適化が実現できる。

次に機能選択項目について他の指定方法を説明する。

【0049】

図12は、図2に示した全体制御部110の仕様解析部106とオブジェクト指向機能排除部107の部分での処理を示したものである。これは、仕様解析部106でオブジェクト名、メソッド名が得られた際に、機能選択項目記述部113を起動させて、得られたオブジェクト名、メソッド名に対応付けて機能選択項目を入力するものである。このようにすれば、機能選択項目の入力パターンにオペレータが入力する必要が無くなる。従って、オペレータには、図13に示すような機能選択項目を表示すればよい。

【0050】

図13に示した機能選択項目は、機能の欄と設定の欄から構成され、機能の欄にはオブジェクト指向の機能の一覧を表示し、オペレータは設定の欄で選択的に

機能の使用／不使用を選べばよい。そして、機能選択記述部 113 はオペレータの入力により、図 14 に示すテーブルを生成し、記憶装置 104 に格納する。尚、各機能毎に“使用”に対するコード生成パターン、“不使用”に対するコード生成パターンは、予め記憶装置 104 に格納しておく。

【0051】

そしてオブジェクト指向機能排除部 107 では、生成されたテーブルを参照し、該当するコード生成パターンを読み出して、コード生成部 108 に出力することにより、コード生成を行う。

【0052】

更に他の選択項目の指定を行う場合について説明する。

【0053】

図 15 は、図 1 で示した入力画面制御部 109 の構成を示したものであり、最適化情報抽出部 1501，解析結果表示部 1502 が新たな構成要素となっている。

【0054】

図 16 は、図 15 に示した構成における使用解析部 106 とオブジェクト指向機能排除部 107 の間の全体制御部 110 の動作を示したものである。この図に示すように仕様解析部 106 で得られた情報から、機能指定に有用な情報を最適化情報抽出部 1501 で抽出し、解析結果表示部 1502 でオペレータに機能指定に有用な情報を表示し、オペレータはその情報をもとに使用する機能を機能選択項目記述部 113 により指定する。最適化情報抽出部 1501 では、例えば使用解析部 106 で解析した結果から得られるインスタンスの数を計数したり、仮想関数の使用の有無を最適化情報として抽出する。この最適化情報を解析結果表示部 1502 により、表示装置 102 に表示する。

【0055】

図 17 (a) に解析結果表示部 1502 による表示例を示す。このように最適化情報抽出部 1501 により得られたインスタンスの数、仮想関数等の最適化情報を表示する。

【0056】

この情報を表示した後に、機能選択項目記述部 113 では図 17 (b) に示す機能選択項目を表示し、オペレータが機能の“使用”，“不使用”を入力する。

このようにオペレータに仕様解析部 106 で解析した結果を利用することで、ユーザが項目を設定する際により有効な設定を行うことができる。例えば、インスタンスの動的生成の使用／不使用はプログラムの中で利用されるインスタンスの数が大きな判断基準であり、オペレータにとって有効な設定を行うことができる。

【0057】

尚、本実施例はプログラミング言語が特に限られるわけではなく C++ や Java で記述したものについても同様に行うことができる。

【0058】

【発明の効果】

以上説明したように、本発明によればオブジェクト指向の使用書からコードを生成する際に、オブジェクト指向プログラミング言語の不要な機能を排除することで、この機能を実現するための機構そのものが削減される。その結果、必要なメモリ容量が大幅に減少されたコードが生成されるようになる。

【図面の簡単な説明】

【図 1】

コード生成装置の全体構成を示した図である。

【図 2】

処理装置の全体制御部の動作を示した図である。

【図 3】

モデル図の入力例を示した図である。

【図 4】

動作図の入力例を示した図である。

【図 5】

機能選択項目の表示例を示した図である。

【図 6】

仕様解析部の動作を示した図である。

【図 7】

オブジェクト指向機能排除部の動作を示した図である。

【図 8】

コード生成部の動作を示した図である。

【図 9】

コードの生成結果を示した図である。

【図 10】

実行コードの生成結果を示した図である。

【図 11】

仮想関数を実行した場合のメモリ配置を示した図である。

【図 12】

全体制御部の動作を示した図である。

【図 13】

機能選択項目の表示例を示した図である。

【図 14】

機能選択項目のテーブルを示した図である。

【図 15】

入力画面制御部の構成を示した図である。

【図 16】

全体制御部の動作を示した図である。

【図 17】

表示例を示した図である。

【符号の説明】

101…入力装置、102…表示装置、103…書込／読出装置、104…記憶装置、105…処理装置、106…仕様解析部、107…オブジェクト指向機能排除部、108…コード生成部、109…入力画面制御部、110…全体制御部、111…クラスインターフェイス記述部、112…処理詳細記述部、113

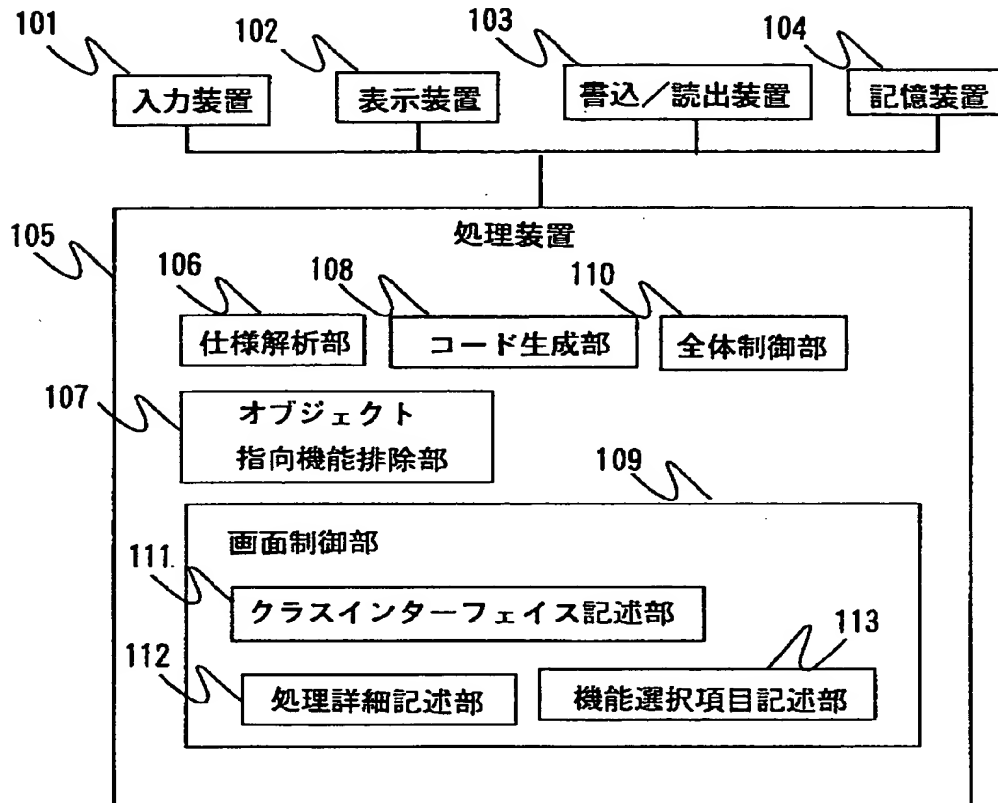
特平 10-038329

…機能選択項目記述部。

【書類名】図面

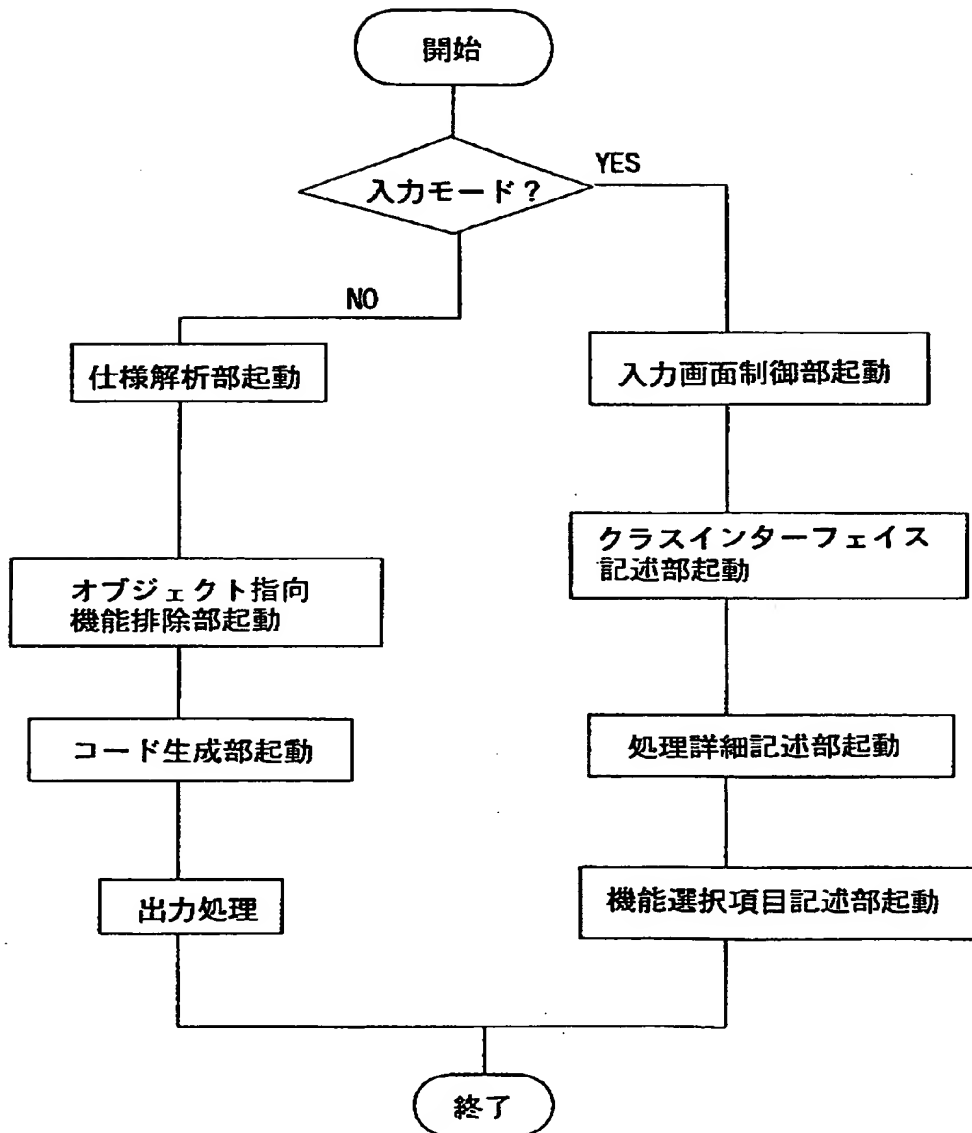
【図 1】

図 1



【図 2】

図 2



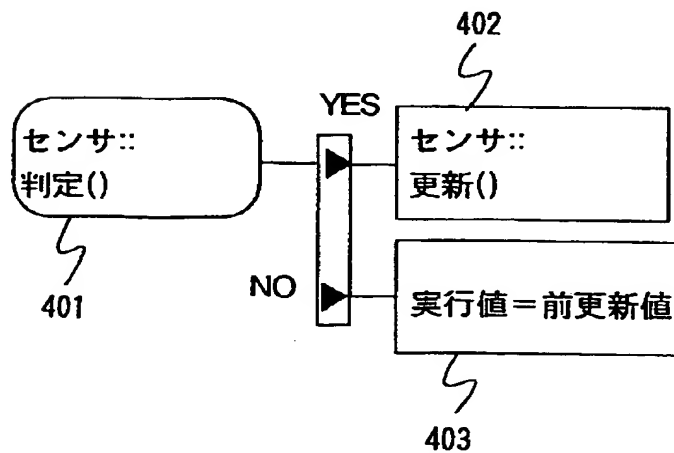
【図 3】

図 3

センサ		301
A/D 変換値	unsigned char	302
更新値	signed short	
前回更新値	signed short	
判定()	boolean : void	303
更新()	void: void	

【図 4】

図 4



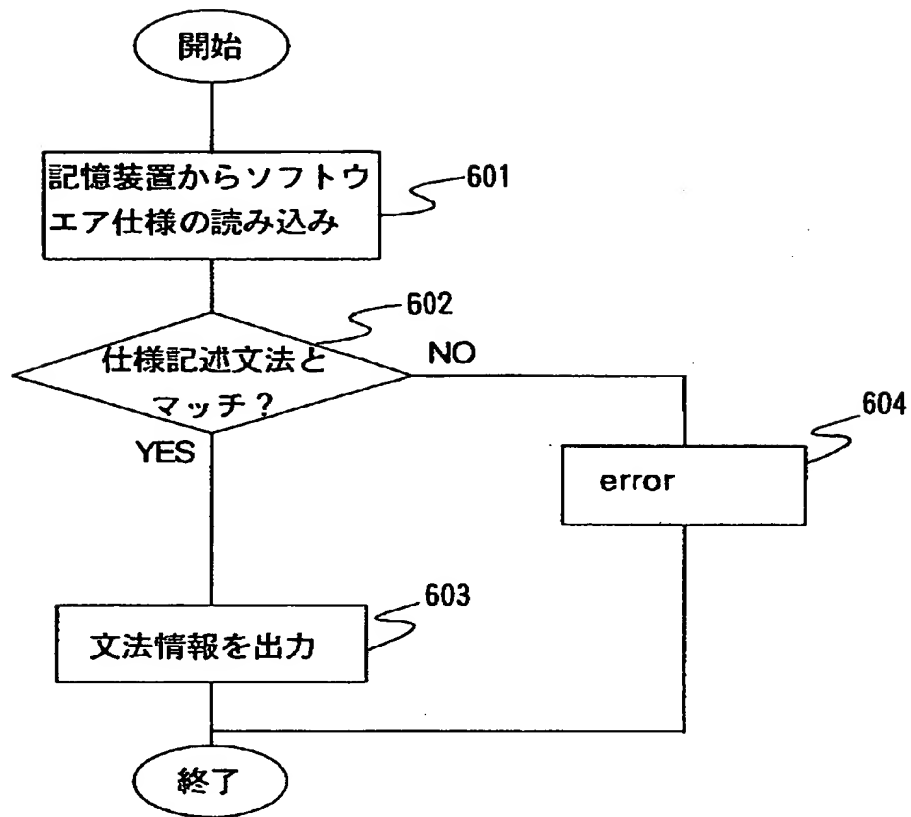
【図 5】

図 5

機能選択項目 <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	
501 入力パターン	<div> オブジェクト名 メソッド名 (引数) </div>
502 使用機能項目	インスタンスの動的生成
503 設定選択肢	使う/ 使わない
504 出力コード	インスタンスの動的生成使う場合 オブジェクト名.メソッド名(引数) インスタンスの動的生成使わない場合 オブジェクト名_メソッド名(引数)

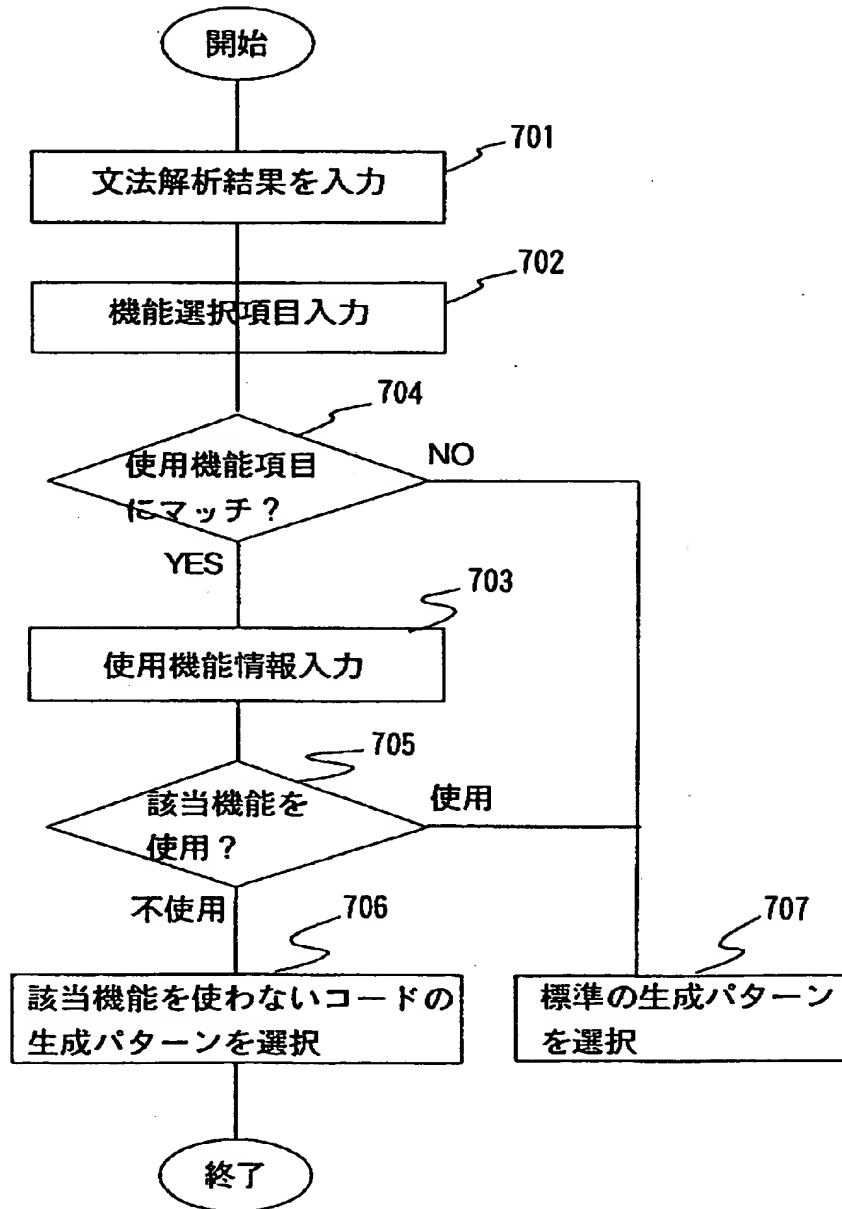
【図 6】

図 6



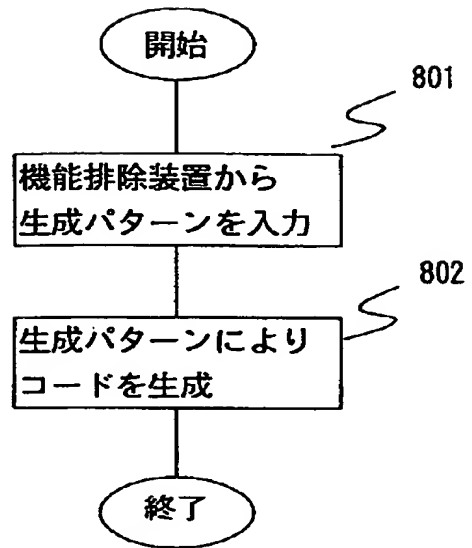
【図 7】

図 7



【図 8】

図 8



【図9】

図 9

```
if(センサ__判定{
    センサ__更新():
}else{
    更新値=前回更新値):
}
```

```
センサ__判定(void){
    ...
}
```

```
センサ__更新(void){
    ...
}
```

(a)動的生成の機能を不使用
(C言語)

```
if(センサ.判定{
    センサ.更新():
}else{
    更新値=前回更新値):
}
```

```
class センサ{
public:
    Bool 判定():
    void 更新():
}
```

```
センサ::判定(void){
    ...
}
```

```
センサ::更新(void){
    ...
}
```

(b)動的生成の機能を使用
(C++言語)

【図 10】

図 10

```
pushl $20
call センサ_判定
addl $4,%esp
movl %eax,%eax
testl %eax,%eax
je .L2
```

(a)動的生成不使用

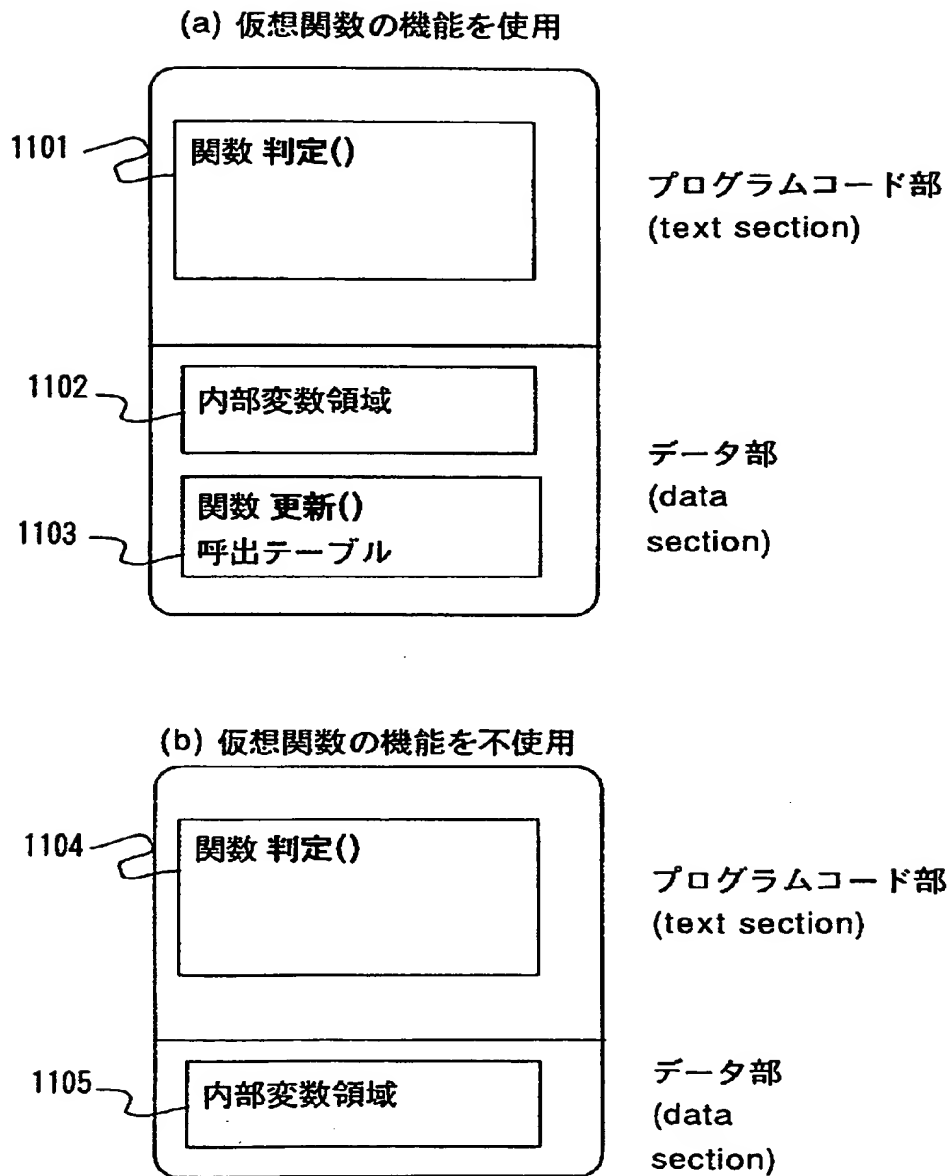
```
pushl $20
leal -88(%ebp),%eax
pushl %eax
call 判定_3 センサ ;
addl $8,%esp
movl %eax,%eax
testl %eax,%eax
je .L238
```

1001

(b)動的生成使用

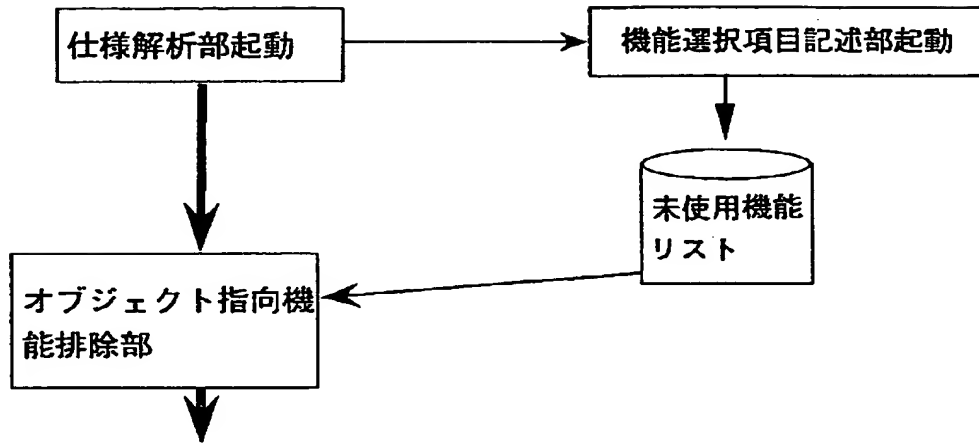
【図 11】

図 11



【図 12】

図 12



【図 13】

図 13

機能排除規則		
装置A用ルール		
機能	設定	
1301 インスタンスの動的生成	<input type="radio"/> 使用	<input checked="" type="radio"/> 不使用
1302 継承	<input checked="" type="radio"/> 使用	<input type="radio"/> 不使用
	⋮	

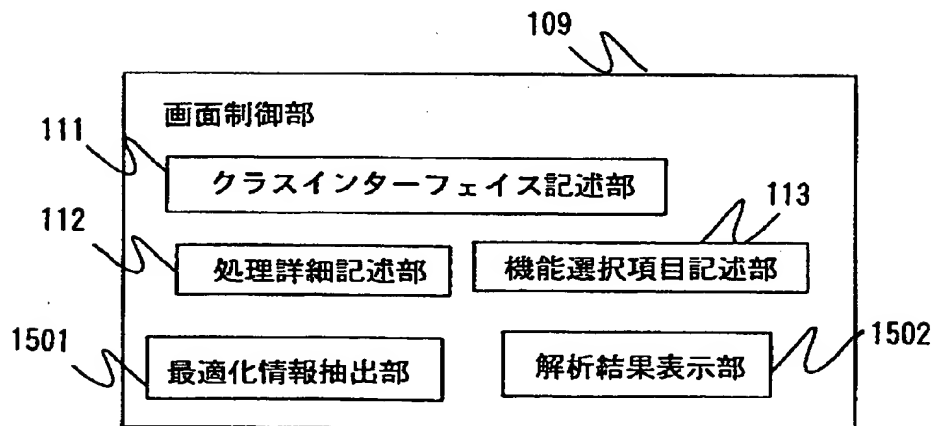
【図 14】

図 14

機能	設定
仮想関数 インスタンスの 動的生成	不使用
	使用
	⋮

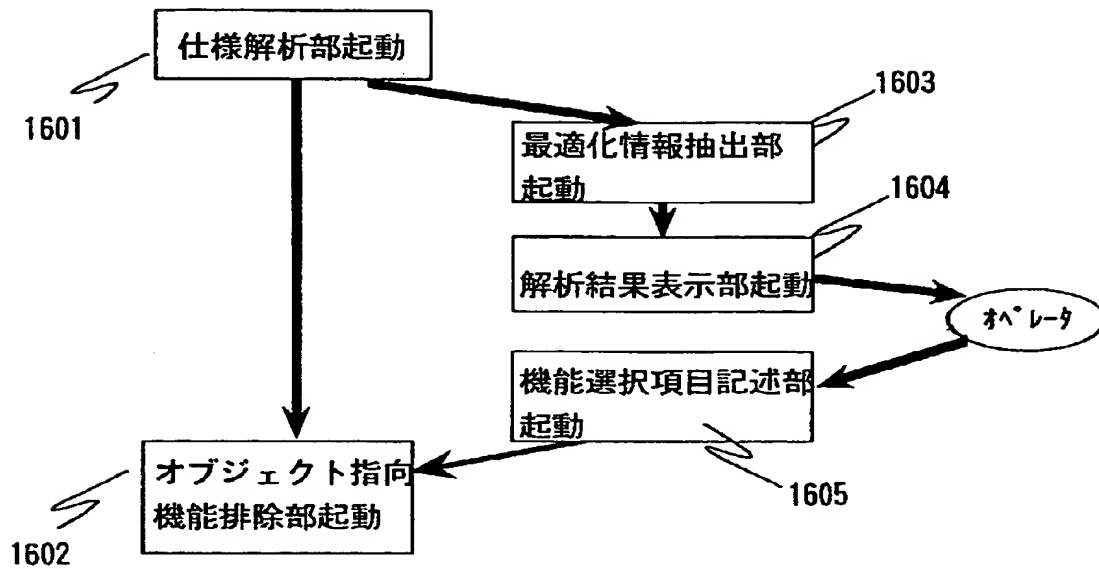
【図 15】

図 15



【図 16】

図 16



【図 17】

図 17

(a) 解析結果表示装置

クラスB

インスタンスの数

親クラス

子クラス

使用している継承メソッド

A::Meth1()

A::Meth3()

未使用メソッド

B::Meth5()

(b) 使用機能指定装置

最適化ルール	
機能	設定
動的な実体化	○ 使用 ● 不使用
継承	○ 使用 ● 不使用
	⋮
定型処理名	Get() Set() ChkStatus()
⋮	⋮

【書類名】 要約書

【要約】

【課題】

必要なメモリ容量を増加させることなく組み込み制御システムに適用可能なコードの最適化を行うコード生成装置を提供すること。

【解決手段】

全体制御部 110 は、記憶装置 104 に記憶されたソフトウェア仕様であるモデル図、動作図を読みだし、仕様解析部 106 を起動させて、字句解析や文法解析を行う。次にオブジェクト指向機能排除部 107 を起動させて、記憶装置 104 に記憶された機能選択項目に基づいて、仕様しない機能を決定する。次に、コード生成部 108 を起動させて、仕様解析部 106 で字句解析、文法解析が行われたソフトウェア仕様とオブジェクト指向機能排除部 107 で決定された出力コードパターンに基づいてコード生成を行う。

【選択図】 図 1

【書類名】 職権訂正データ
【訂正書類】 特許願

<認定情報・付加情報>

【特許出願人】
【識別番号】 000005108
【住所又は居所】 東京都千代田区神田駿河台四丁目6番地
【氏名又は名称】 株式会社日立製作所
【代理人】 申請人
【識別番号】 100068504
【住所又は居所】 東京都千代田区丸の内1-5-1 株式会社日立製
作所 知的所有権本部内
【氏名又は名称】 小川 勝男

出 願 人 履 歴 情 報

識別番号 [000005108]

1. 変更年月日 1990年 8月31日
[変更理由] 新規登録
住 所 東京都千代田区神田駿河台4丁目6番地
氏 名 株式会社日立製作所